From: Hans Reiser
To: Microsoft ATR
Date: 1/28/02 11:03am
Subject: Microsoft Settlement

[Text body exceeds maximum size of message body (8192 bytes). It has been converted to attachment.]

CC: reiser@namesys.com@inetgw

If you are not able to process html format for proper printing, or you lost the html version I sent, please accept this email (excepting this sentence) as my comment on the proposed settlement, otherwise please accept the html version which preceded this.

MS Settlement Reflects Deep Failure To Understand Implications of "Patching" Technology

The positions of the DOJ, the States, and even Lawrence Lessig are based on a failure to understand that something unique to the software industry, which programmers call "patching" technology, makes software products infinitely separable if an essential facility called "source code" is provided. No disclosure of APIs, and no structuring of APIs, can accomodate all potential products in the manner that disclosure of source code plus use of patching does. Every line of source code is a possible location for insertion of new code that forms a new product. This new code can be distributed separately from the original source code, and post-sale added by the consumer, via what programmers call a "patch". Patching technology fundamentally changes product separability, making separation dependent on the essential facility called "source code". Non-programmers seem to not yet understand this. Persons who work in the Linux industry know this from experience, and I will try to convey this experience as someone who has built a business from the sale of patches (for the ReiserFS filesystem) in the only market where I had access to kernel source code.

Software is unique in that "Compiler" technology allows consumers to effectively reassemble software themselves.

A compiler is a computer program that takes a set of instructions about how to build a program (called "source code"), and builds the software. Almost all software is actually assembled by compilers not humans, and the work of humans is almost entirely in creating the source code.

You have probably never used a compiler to assemble software yourself as a consumer because:

* you are not a Fortune 500 company with a staff of trained system administrators
* you probably use Windows not Linux, and Windows does not give you access to the essential facility known as "source code" that your "compiler" needs to reassemble your software
* the new crusade by Linux to make the compilation process user friendly has only just started

Because you have never done it yourself, your intuition may tell you that it is not feasible, or that it is not feasible for a large market. Beware this intuition, it is simply wrong. The Fortune 500 are a significant market for antitrust purposes, and Linux is rapidly moving towards making asking compilers to perform reassembly a friendly experience for average persons.

It is frequently efficient to post-sale integrate software for a large part of the market, and it is getting more so with time. This is deeply different from physical products such as cars, in that most persons do not find it as effective to buy a collection of parts and self-assemble because they would have to do the work of assembly. With software, the computer does the work of post-sale assembly, and the consumer simply tells the computer to do it, goes to make some tea, comes back, and the job is done.

For instance, the business that I own (Namesys, see www.namesys.com) made its money entirely from sales of a filesystem (ReiserFS) that was sold separately from the operating system (Linux) for the first few years of our business. The revenues from this were enough to support us. Paying consumers such as MP3.com would take our source code, add it to the Linux kernel source code, use a compiler, let their computer do a few minutes of work to reassemble the kernel, and get a better filesystem as a result of it. This allowed MP3.com to save $20 million dollars according to their estimate. Others in my industry also sell

filesystems separately from operating systems (www.veritas.com got its start that way, and still makes simply enormous amounts of money from doing so, there are others....).

Notice that I say filesystem. Your intuitive notion of what is an operating system probably tells you that the filesystem is part of the operating system. You may be tempted to think that what is part of the operating system is not viable as a product sold separately from the operating system. Lessig thought so, and this is because he lacks experience selling operating system components in the Linux/Unix programming industry.

Think of Jefferson Parish, and understand that software takes the fine distinctions of Jefferson Parish to their extreme:

* Software can be integrated in its functioning, and yet separate in its sale, and this means separate as a product for purposes of anti-trust law. (Most software products are functionally integrated with a separately sold operating system.)
* Software can be integrated in its physical distribution, yet separate in its sale. (Purchase of a CDROM holding the software is often separated from purchase of a license to use, and it is often considered efficient by publishers to bundle physical distribution without bundling licensing.)
* Software can be sold and transmitted over the Internet with no physical product created at all.

There is only one characteristic that necessarily defines the separation of a software product, and that is the license. A license is a contract, and contractual tying is illegal under the Clayton and Sherman acts.

Yet wait, if software products are so easily separable, why aren't there far more OS components out there being sold? Control over an essential facility is the answer.

Secret source code can be an essential facility the equal of putting a combination lock on every bolt in a car, and then declaring the combination to be a trade secret.

You wouldn't allow this for a car, yet traditional industry practice is that source code is kept a trade secret. The crisis our industry is facing, in which monopoly control is the norm in all parts of it not in infancy, is directly caused by this industry practice of secret source code. It is not necessary that the text be kept secret for copyright protection on books to be maintained, and it is also not necessary for software that the source be kept secret to protect ownership of it. Far from it, the underlying historical motivation of copyright and patent laws is to bring more information out of trade secret status.

We have a widespread well-entrenched industry practice that keeps an essential facility (source code) under the control of monopolists (of which Microsoft is merely the largest), and we have almost complete monopolization of the software industry in each of its mature niches. These are cause and effect.
I pray to you to not allow their continuance. Open up the operating system source code, and go even further. Declare that software is per se separable where source code is available. Declare source code to be an essential facility. Return copyright and patent practices to their historical roots, and require that information created be made public if it is to be protected.
Please do not hesitate to ask me to comment in greater detail or respond to your questions in this matter. I am available for in person testimony if desired.

I have great respect for Reilly and Lawrence Lessig generally, and for their arguments in most other matters, and I hope it is understood that I merely have an advantage in possessing "patches" sales experience.

As for my needs, please create the legal conditions which will allow me to port ReiserFS to Windows and sell it separately from the operating system, by giving me the access to source code that I need to do the port, and to sell the patch

separately from the OS.

Essential Facilites Related Citations

[U.S. vs. Trans-Missouri Freight Ass'n, 166 U.S. 290 (1897)] is the original precedent.

[MCI Communications v. AT&T Corp. 708 F.2d 1081 (7th Cir.), cert. denied, 464 U.S. 891 (1983)] describes a case more recent (it is a persuasive rather than controlling authority). Note that the 4 part test lacks any component referencing the need for a market to have been active at some point prior to the refusal to deal, and is the better for that lack.

Profit To The Monopolist From Tying

The Chicago School, to which the current DOJ administration adheres, holds that there is no incentive to monopolists to engage in tying because it believes they cannot extract more profit from forced sales of the tied product than they would from raising the price of the tying product, unless business efficiencies exist. For this reason, they feel that there is no need for the Clayton prohibition against tying, and feel there are civil liberty reasons to avoid government intervention into markets. Their analysis assumes the tied product is part of a fully competitive market, and for this reason it is deeply flawed. The profit to the monopolist from engaging in tying is the difference between the market price and the marginal cost. For less than fully competitive markets, which is to say most markets, this is a non-zero amount. For software, especially software sold and distributed over the Internet, the marginal cost is close to zero, and the motivation for engaging in tying is extremely high. Senators Sherman and Clayton were much more knowledgable about economics than the Chicago School is paid to think (various monopolists have given large funding sums to pro-trust law schools). Some might like to think that, but for government, free choice expressed in the market would free us, but in sad reality the government is not the only means by which people organize to control and plunder the public. Cartels and monopolies take away our freedoms as well. The only thing worse than a government controlled economy is a monopoly controlled economy.

The Settlement As A Whole

I am opposed to the settlement as a whole. President Bush owns stock in Microsoft, and he appointed to head the antitrust division at the DOJ someone who is widely known to be opposed to laws against tying. When someone is opposed to a law that they are supposed to prosecute, they should not be allowed to settle a case their predecessor started. The proposed settlement is designed to be toothless, and to do nothing. Do not allow President Bush to settle this case, and thereby cripple the ability of the next administration to enforce the law. The failure of Microsoft and the DOJ to adhere to the contact disclosure provisions of the Tunney Act is one more reason to reject the settlement.

Conclusion

If you have the courage to firmly reject this settlement, if you declare software to be per se separable, and if you move aggressively to enforce the claim of the States while we wait for a new administration, you will have earned the admiration of the American people. Some of them will even know this. More importantly, you will.

Sincerely,

Hans Reiser
Owner/Operator of Namesys

Author of ReiserFS, a significant component of Linux

5918 Marden Lane
Oakland, CA 94611
phone: +1 510-339-1044 (USA)
+7 095 290 6405 (I am currently in Russia)